

METHOD, SYSTEM, AND ARTICLE OF MANUFACTURE FOR GENERATING
DEVICE SPECIFIC REQUESTS

BACKGROUND

5 1. Field

[0001] The disclosure relates to a method, apparatus, system, and article of manufacture for generating device specific requests.

2. Background

10 [0002] The Common Information Model (CIM) is an industry standard specification to allow for the interchange of management information in a network environment including devices from different vendors, where the network may include heterogenous as well as homogeneous devices. The CIM schema specifies a set of classes, including methods and objects, that management programs call to obtain information and perform
15 management operations with respect to devices in the network. Each vendor of a network device that is capable of interfacing in a CIM environment may provide a set of device specific APIs that implement the CIM classes. A vendor would provide a CIM Provider, which is a program module that maps CIM APIs or methods, as defined by the industry standard CIM model, to device specific APIs that may implement the
20 functionality of a defined CIM API for the specific device. The term "CIM API" as used herein refers to any method, interface or function that is called to perform an operation defined within the CIM management schema.

[0003] The CIM schemas define a common set of models and semantics for the management of devices. Applications can be programmed against a known, consistent set
25 of models. For programming such applications, there is a continued need in the art to provide device vendors improved techniques to develop CIMs that convert device independent CIM requests to device specific requests for managed devices.

SUMMARY OF THE DESCRIBED EMBODIMENTS

[0004] Provided are a method, system and article of manufacture for managing devices, wherein in certain embodiments a request implemented via at least one device independent class is received. A class hierarchy database is traversed to determine at least one device specific class that corresponds to the at least one device independent class, wherein the class hierarchy database stores a class hierarchy and associations between classes. The received request is modified, wherein in the modified request the least one device independent class has been translated to the at least one device specific class.

5 [0005] In certain additional embodiments, at least one device independent class attribute is mapped to at least one device specific class attribute in the modified request. At least one device independent property is mapped to at least one device specific property in the modified request. A device specific request is generated from the modified request, in response to mapping the at least one device independent class attribute and the at least

10 15 one device independent property. The device specific request is sent to a managed device.

[0006] In further embodiments, the received request is further modified to include at least one association between device specific classes in the class hierarchy.

20 [0007] In yet additional embodiments, the received request indicates a source class and a requested class. A specific association is determined between a first device specific class that corresponds to the source class and a second device specific class that corresponds to the specific class, wherein the specific association corresponds to a managed device. In certain embodiments, the source class represents storage pools and the requested class represents storage volumes corresponding to a storage pool.

25 [0008] In still further embodiments, the received request indicates a source class and a base association. A first device specific class is determined from the class hierarchy database, wherein the first device specific class has a specific association with a second device specific class that corresponds to the indicated source class, and wherein the specific association corresponds to the base association.

[0009] In yet additional embodiments the receiving, traversing, and modifying are performed by a proxy. A device specific request is generated in a device specific language. The device specific request in the device specific language is sent to a managed device coupled to the proxy.

5 [0010] In additional embodiments, the request is received from a Common Information Model application, wherein the at least one device independent class is specified by a Common Information Model schema.

[0011] In further embodiments, the request comprises a command that is part of an object oriented management schema for managing non-homogeneous devices in a network

10 environment. In certain embodiments the management schema comprises the Common Information Model.

BRIEF DESCRIPTION OF THE DRAWINGS

[0012] Referring now to the drawings in which like reference numbers represent 15 corresponding parts throughout:

FIG. 1 illustrates a computing environment, in accordance with certain embodiments;

FIG. 2 illustrates how a CIM request is converted to a device specific request, in accordance with certain embodiments;

20 FIG. 3 illustrates logic to generate one or more device specific requests, in accordance with certain embodiments;

FIGs. 4, 5, 6, 7 illustrate how device specific CIM classes and associations are generated from CIM classes, in accordance with certain embodiments;

25 FIG. 8 illustrates how a mapping application generates device specific requests in a device specific language, in accordance with certain embodiments;

FIG. 9 illustrates how attributes and properties are mapped by the mapping application, in accordance with certain embodiments; and

FIG. 10 illustrates a computing architecture in which certain embodiments are implemented.

DETAILED DESCRIPTION

[0013] In the following description, reference is made to the accompanying drawings which form a part hereof and which illustrate several embodiments. It is understood that other embodiments may be utilized and structural and operational changes may be made.

[0014] FIG. 1 illustrates a computing architecture in which aspects of the invention are implemented. A plurality of host systems 102a, 102b,...102n, a CIM Object Manager (CIMOM) 104, CIM providers 106a...106m, and managed devices 108a, 108b,...108p communicate over a network 110. In certain embodiments the CIMOM 104 and the CIM providers 106a...106m may be implemented in a proxy 112, where the proxy 112 may be a computational device. Each host 102a...102n includes a CIM application 110a, 110b...110n to generate and communicate CIM management requests comprised of CIM APIs to perform management operations with respect to the managed devices 108a...108p. The CIMOM 104 receives CIM requests from the CIM applications 110a...110n and transfers them to the CIM provider 106a...106m associated with the managed device 108a...108p to which the request is directed. Each managed device 108a...108p implements device specific APIs 114a...114p which perform management related operations, retrieving information, configuration, etc., on the device 108a...108p.

20 A device specific API, such as, device specific API 114a, that is implemented in a managed device, such as, managed device 108a, may use one or more device specific classes corresponding to the managed device.

[0015] The CIMOM 104 may include a class hierarchy database 116 that maps CIM classes, such as, standard or Distributed Management Task Force (DTMF) CIM classes, to device specific CIM classes by storing classes hierarchically. While in certain embodiments CIM classes may comprise standard or DTMF CIM classes, in alternative embodiments CIM classes may comprise other classes that are not device specific. For example, CIM classes may include device independent classes designed by an organization that is different from DTMF. A CIM provider, such as CIM provider 106a,

may include a traversal application 118, a mapping application 120, and a mapping store 122. The traversal application 118 may traverse the class hierarchy database 116 to map CIM classes present in a request from a CIM application 110a...110n to device specific CIM classes. The mapping application 120 may use the mapping store 122 to store a 5 class hierarchy that contains CIM classes, such as, standard or DTMF CIM classes, and device specific CIM classes. In certain embodiments, the CIM classes are super-classes and the device specific classes are sub-classes or leaf-classes in the stored class hierarchy.

[0016] The CIM providers 106a...106m are used to map CIM commands in CIM 10 messages to device specific APIs 114a, 114b,..114p capable of implementing the CIM command on the target managed device 108a...108p. Further details of the CIM model are described in publications from the Distributed Management Task Force (DMTF), including "Common Information Model: Core Model", Version 2.4 (August 30, 2000), which publication is incorporated herein by reference in its entirety.

[0017] The network 110 may comprise any network known in the art, such as a Local 15 Area Network (LAN), Storage Area Network (SAN), Wide Area Network (WAN), the Internet, a wireless network, etc. Alternatively, the network 110 may comprise a bus interface. The hosts 102a...102n may comprise any type of computational device capable of running a CIM application 110a...110n, such as a workstation, desktop 20 computer, server, laptop, mainframe, telephony device, hand held computer, etc. The proxy 112 may comprise any type of computational device capable of running the CIMOM 104 and the CIM providers 106a...106m. In alternative embodiments, the CIM providers 106a...106m may run on systems separate from the CIMOM 104 or run within the managed devices 108a...108p. Further, one CIM provider 106a...106m may manage 25 CIM messages for one or more managed devices 108a...108p. The managed device 108a...108p may comprise any physical or logical device known in the art, such as a storage device, storage medium, storage media library, Fibre Channel, switch, fabric, database, etc., for which a separate CIM provider may be provided. There may be any

number of hosts, CIMOMs, CIM providers, and managed devices, and embodiments are not limited to the configuration and arrangement of components shown in FIG. 1.

[0018] FIG. 2 illustrates how the proxy 112 converts a CIM request 200 into one or more device specific requests 202a...202q, in accordance with certain embodiments. Device 5 specific requests are implemented using device specific classes, i.e., a device specific request corresponding to a managed device can be executed in the managed device.

[0019] The proxy 112 receives a CIM request 200 that uses one or more CIM classes from a CIM application, such as, CIM application 110a that runs in the host 102a. In certain embodiments, the CIM request 200 is intended for the managed device 108a 10 which is managed by the CIM provider 106a in the proxy 112. The proxy 112 modifies the CIM request 200 into at least one modified request 204 that uses one or more device specific CIM classes. The proxy 112 further converts the modified request 204 into one or more device specific requests 202a...202q that uses device specific language and sends the one or more device specific requests 202a...202q to a managed device, such as 15 managed device 108a, whose device API 114a uses the device specific language.

[0020] Therefore, FIG. 2 illustrates an embodiment in which the proxy 112 converts a CIM request 200 received from a CIM Application 110a into one or more device specific requests 202a...202q for the managed device 108a. The proxy 112 may convert CIM requests from other CIM applications into device specific requests for other managed 20 devices.

[0021] FIG. 3 illustrates logic implemented in the proxy 112 to generate one or more device specific requests 202a...202q, in accordance with certain embodiments. In alternative embodiments, certain of the operations illustrated in FIG. 3 may be performed in a different order, modified or removed.

25 [0022] Control starts at block 300 where the proxy 112 receives a CIM request 200 from a CIM application, such as, CIM application 110a, where the CIM request 200 is implemented via CIM classes. In certain embodiments, the CIM request 200 is intended for the managed device 108a, where the managed device 108 is managed by the CIM

provider 106a in the proxy 112. In other embodiments, the CIM request 200 may be intended for other managed devices that may be managed by other CIM providers.

[0023] The traversal application 118 in the CIM Provider 106a of the proxy 112 traverses the class hierarchy database 116 in the CIMOM 104 to determine (at block 302) all 5 specific subclasses of the CIM superclass that has a specific class corresponding to the managed device for which the CIM requested is intended. The CIM superclass is the class from which all other CIM classes derive.

[0024] The traversal application 118 determines (at block 304) whether the CIM request 200 had requested associations between device specific CIM classes to be derived. In 10 objected oriented terminology an association is a connection between two classes.

[0025] If the traversal application 118 determines that the CIM request 200 had requested associations between device specific classes to be derived, then the traversal application 118 on the CIM provider 106a generates (at block 306) the associations. The traversal application 118 generates (at block 308) the modified request 204 using device specific 15 CIM classes and optionally uses the associations.

[0026] The traversal application 118 forwards (at block 310) the modified request 204 that includes the device specific classes and optionally the associations to the mapping application 120 in the CIM provider 106a. The mapping application 120 in the CIM provider 106a uses the mapping store 122 to map (at block 312) CIM class attributes to 20 device class attributes and CIM properties to device properties to generate one or more device specific requests 202a...202q. The mapping application 120 sends (at block 314) the one or more device specific requests 202a...202q in the device specific language to the managed device 108a.

[0027] If the traversal application 118 determines (at block 304) that the CIM request 25 200 had not requested associations between device specific classes to be derived, then the traversal application 118 on the CIM provider 106a generates (at block 308) the modified request 204 using device specific CIM classes.

[0028] Therefore, the logic of FIG. 3 illustrates certain embodiments in which the CIM provider 106a converts the device independent CIM request 200 from the CIM

application 110a into one or more device specific requests 202a...202q for the managed device 8a.

[0029] FIGs. 4, 5, 6, 7 illustrate how device specific CIM classes and associations are generated from CIM classes by logic implemented in the proxy 112, in accordance with 5 certain embodiments.

[0030] FIG. 4 illustrates an embodiment in which a class hierarchy 400 is stored in the class hierarchy database 116. The bases classes and base associations are with respect to CIM classes and the specific classes and specific associations are with respect to the device specific CIM classes. In certain embodiments, if a requested class 402 from the 10 traversal application 118 is the base class A 404, then in certain embodiments the corresponding returned classes 406 to the traversal application 118 from the class hierarchy database 116 may be the specific class A 408 and the specific class B 410.

[0031] FIG. 5 illustrates an embodiment in which a class hierarchy 500 is stored in the class hierarchy database 116. The bases classes and base associations are with respect to 15 CIM classes and the specific classes and specific associations are with respect to the device specific CIM classes. In certain embodiments if a requested class 502 from the traversal application 118 is the base association 504, then in certain embodiments the corresponding returned class 506 to the traversal application 118 from the class hierarchy database 116 may include the specific association 508.

[0032] FIG. 6 illustrates an embodiment in which a class hierarchy 600 is stored in the class hierarchy database 116. The bases classes and base associations are with respect to CIM classes and the specific classes and specific associations are with respect to the device specific CIM classes. In certain embodiments if the traversal application 118 20 requests specific association instances by providing a source class 602 and a requested class 604, then the traversal application 118 may derive the class supported by the device 614 as the specific association 606. For example, in certain embodiments the base class B 608 may represent a storage pool and the base class C 610 may represent storage volumes 25 in the storage pool, where the base association 612 connects the base class B 608 to the base class C 610. In certain embodiments, the traversal application 118 may determine

the class supported by the device 614 from the class hierarchy 600, where the specific association 606 associates the specific class A 616 to the specific class B 618.

[0033] FIG. 7 illustrates an embodiment in which a class hierarchy 700 is stored in the class hierarchy database 116. The bases classes and base associations are with respect to

5 CIM classes and the specific classes and specific associations are with respect to the device specific CIM classes. In certain embodiments, the traversal application 118 provides a source class 702 and requests the class hierarchy database 116 to provide the requested class 704 that corresponds to the class associated with the source class. For example, in certain embodiments the base class B 706 may represent a storage pool and
10 the base class C 708 may represent storage volumes in the storage pool, where the base association 710 connects the base class B 706 to the base class C 708. In certain embodiments, the class hierarchy database 116 may return a class supported by the device 712 as the specific class B 714, where the specific association 716 associates the specific class 714 to the specific class 718 that corresponds to the base class B 706, i.e.,
15 the source class 702.

[0034] Therefore, FIGs. 4-7 illustrate how classes and associations may be derived by the traversal application 118 from the class hierarchy database 116.

[0035] FIG. 8 illustrates how a mapping application, such as the mapping application 120, implemented in a CIM provider, such as, the CIM provider 106a, generates device 20 specific requests 202a...202q in a device specific language, in accordance with certain embodiments.

[0036] In certain embodiments, the mapping application 120 takes the modified request 204 generated by the traversal application 118 as input and in association with the mapping store 122 generates one or more device specific requests 202a...202q, where the 25 device specific requests 202a...202q use device specific language corresponding to a device, such as, the managed device 108a, for which the modified request 204 is intended. The mapping store 222 includes mappings of CIM class attributes to device attributes and mappings of CIM properties to device properties. The device specific

requests 202a...202q are implemented in the device specific language and include the device attributes and the device properties.

[0037] For example, if the traversal application 118 generated the modified request 204 from the CIM request 200, then in certain embodiments the mapping application 220 may 5 use the modified request 204 as input to generate the device specific requests 202a...202q for the managed device 8a.

[0038] FIG. 9 illustrates how attributes and properties, such as, data types, are mapped by the mapping application 220 implemented in the CIM provider 106a in the proxy 112, in accordance with certain embodiments.

10 [0039] The mapping store 222 may include data structures corresponding to an attribute mapper 900 and a property mapper 902, such as, a mapper for data types. The attribute mapper 900 includes mappings between CIM attributes of an object and device attributes of the object. The property mapper 902 includes mappings between CIM properties of an object and device properties of an object. For example, in certain embodiments, the 15 property mapper 902 may map data types.

[0040] In certain embodiments, the mapping application 220 may take a CIM class 904 with CIM property 906 and inputs and generate the device class 908 with the corresponding device property 910. The attributes of the CIM class 904 are mapped to the device attributes of the device class 908 from the attribute mapper 900 and the 20 corresponding properties, such as, data types, are mapped by the property mapper 902.

[0041] Therefore, FIG. 9 illustrates how the mapping application 120 in a CIM provider in the proxy 112 generates device specific requests in a device specific language by mapping attributes and properties, such as, data types.

[0042] Certain embodiments inspect the CIM class definitions and a class hierarchy 25 database and finds appropriate device specific CIM classes corresponding to device independent CIM classes. The class hierarchy database may be referred to as a dictionary and the dictionary may be configured to different types of class hierarchies and different types of devices. Certain embodiments also determine association between classes. Other

embodiments provide additional generic techniques for the translation of device specific CIM classes to devices specific requests in a device specific language.

Additional Embodiment Details

5 [0043] The described techniques may be implemented as a method, apparatus or article of manufacture involving software, firmware, micro-code, hardware and/or any combination thereof. The term “article of manufacture” as used herein refers to program instructions, code and/or logic implemented in circuitry (e.g., an integrated circuit chip, Programmable Gate Array (PGA), ASIC, etc.) and/or a computer readable medium (e.g.,

10 magnetic storage medium, such as hard disk drive, floppy disk, tape), optical storage (e.g., CD-ROM, DVD-ROM, optical disk, etc.), volatile and non-volatile memory device (e.g., Electrically Erasable Programmable Read Only Memory (EEPROM), Read Only Memory (ROM), Programmable Read Only Memory (PROM), Random Access Memory (RAM), Dynamic Random Access Memory (DRAM), Static Random Access Memory (SRAM), flash, firmware, programmable logic, etc.). Code in the computer readable medium may be accessed and executed by a machine, such as, a processor. In certain embodiments, the code in which embodiments are made may further be accessible through a transmission medium or from a file server via a network. In such cases, the article of manufacture in which the code is implemented may comprise a transmission

15 medium, such as a network transmission line, wireless transmission media, signals propagating through space, radio waves, infrared signals, etc. Of course, those skilled in the art will recognize that many modifications may be made without departing from the scope of the embodiments, and that the article of manufacture may comprise any information bearing medium known in the art. For example, the article of manufacture

20 comprises a storage medium having stored therein instructions that when executed by a machine results in operations being performed.

25 [0044] FIG. 10 illustrates a block diagram of a computer architecture in which certain embodiments are implemented. FIG. 10 illustrates one embodiment of the hosts 102a...102n, and the proxy 112. The hosts 102a...102n and the proxy 112 may implement

a computer architecture 1000 having a processor 1002, a memory 1004 (e.g., a volatile memory device), and storage 1006. Not all elements of the computer architecture 1000 may be found in the hosts 102a...102n and the proxy 112. The storage 1006 may include a non-volatile memory device (e.g., EEPROM, ROM, PROM, RAM, DRAM, SRAM, 5 flash, firmware, programmable logic, etc.), magnetic disk drive, optical disk drive, tape drive, etc. The storage 1006 may comprise an internal storage device, an attached storage device and/or a network accessible storage device. Programs in the storage 1006 may be loaded into the memory 1004 and executed by the processor 1002 in a manner known in the art. The architecture may further include a network card 1008 to enable 10 communication with a network. The architecture may also include at least one input device 1010, such as a keyboard, a touchscreen, a pen, voice-activated input, etc., and at least one output device 1012, such as a display device, a speaker, a printer, etc.

[0045] At least certain of the operations of FIG. 3 may be performed in parallel as well as sequentially. In alternative embodiments, certain of the operations may be performed in a 15 different order, modified or removed.

[0046] Furthermore, many of the software and hardware components have been described in separate modules for purposes of illustration. Such components may be integrated into a fewer number of components or divided into a larger number of components. For example, the traversal application 118 and the mapping application 120 20 could comprise a single application. Additionally, certain operations described as performed by a specific component may be performed by other components.

[0047] The data structures and components shown or referred to in FIGs. 1-10 are described as having specific types of information. In alternative embodiments, the data structures and components may be structured differently and have fewer, more or 25 different fields or different functions than those shown or referred to in the figures.

[0048] Therefore, the foregoing description of the embodiments has been presented for the purposes of illustration and description. It is not intended to be exhaustive or to limit the embodiments to the precise form disclosed. Many modifications and variations are possible in light of the above teaching.